DEBIAN 12 netinst x64

0. Descarga iso:

https://www.debian.org/CD/netinst/

1. Usuarios:

·root -> modemsys1234

·tc -> modemsys1234

2. Al final de la instalación escoger:

·Sin entorno gráfico

·SSH

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\* MANUAL CONFIGURACIÓN DEBIAN 12 PARA TSM \*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-Acceder con usuario root (password modemsys1234)

-Instalar sudo:

apt install sudo

-Instalar:

sudo apt install libffi-dev libssl-dev

-ETH:

0. Comprobar ip:

ip a s

1. Configurar IP

sudo nano /etc/network/interfaces

auto enp0s3

iface enp0s3 inet static

address 192.168.0.35

netmask 255.255.0.0

gateway 192.168.0.1

dns-nameservers 8.8.8.8 8.8.4.4

2. Reiniciar red:

sudo systemctl restart networking

-INSTALAR ifconfig

sudo apt-get update

sudo apt install net-tools

echo $PATH

export PATH="/sbin:/usr/sbin:${PATH}"

ifconfig

/usr/sbin/ifconfig

-CONFIGURAR EL PUERTO ETH COMO ETH0:

0. ip link show 🡪 Buscar la mac del adaptador de red a renombrar

1. sudo nano /etc/udev/rules.d/70-persistent-net.rules

2. Escribir:

SUBSYSTEM=="net", ACTION=="add", ATTR{address}=="xx:xx:xx:xx:xx:xx", NAME="eth0" --> **"xx:xx:xx:xx:xx:xx"=MAC del adaptador a renombrar**

3. sudo nano /etc/network/interfaces

4. Escribir:

auto eth0

iface eth0 inet static

address 192.168.0.45

netmask 255.255.0.0

gateway 192.168.0.1

dns-nameservers 8.8.8.8 8.8.4.4

5. sudo update-initramfs -u

6. sudo reboot

-MQTT (mosquitto-clients)

sudo apt install mosquitto-clients

-SSH: Debería estar instalado con el SO.

0. Comprobar si está instalado:

sudo systemctl status ssh

a) Si no está instalado:

sudo apt update

sudo apt install openssh-server

sudo systemctl enable ssh

b) Si está instalado:

sudo systemctl enable ssh

1 . Configuración acceso user/passwd:

sudo nano /etc/ssh/sshd\_config

--> PasswordAuthentication YES

2. Reiniciar servicio:

sudo systemctl restart ssh

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\* Desde aquí continuar por SSH (copiar/pegar) \*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

-Configurar el grub para arranque directo de SO:

1. Editar config. grub:

sudo nano /etc/default/grub

GRUB\_DEFAULT=0

GRUB\_TIMEOUT=0

2. Actualizar config grub:

sudo update-grub

-Configurar autologin de usuario "tc":

1. Preparar SO:

sudo nano /etc/systemd/logind.conf --> Activar y modificar

NAutoVTs=6

ReserveVT=7

2. Crear servicio:

sudo mkdir /etc/systemd/system/getty@tty1.service.d/

sudo nano /etc/systemd/system/getty@tty1.service.d/override.conf --> Añadir:

[Service]

ExecStart=

ExecStart=-/sbin/agetty --noissue --autologin tc %I $TERM

Type=idle

3. Reiniciar:

sudo reboot

-Modificar usuario tc para sudo sin passwd:

sudo nano /etc/sudoers

·Anadir tras la línea "root ALL=(ALL:ALL) ALL":

tc ALL=(ALL) NOPASSWD: ALL

sudo visudo -c

-MARIADB:

1. Instalación:

sudo apt update

sudo apt install mariadb-server

2. Configuración:

sudo mysql\_secure\_installation ->

(root\_pass: modemsys1234, y,y -> modemsys1234, n,,n, n,, y)

2. Activación:

sudo systemctl enable mariadb

sudo systemctl start mariadb

-HABILITAR ACCESO REMOTO A MARIADB (ejemplo HeidySQL):

1. sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf

2. Modificar:

bind-address = 0.0.0.0

3. sudo systemctl restart mariadb

4. sudo mysql -u root -p

5. GRANT ALL PRIVILEGES ON \*.\* TO 'root'@'%' IDENTIFIED BY 'modemsys1234' WITH GRANT OPTION; FLUSH PRIVILEGES; EXIT;

6. Preparar para abrir puertos  
 sudo apt update

sudo apt install ufw

7. Abrir Puertos:

7.1 mysql

sudo ufw allow 3306-tcp

7.2 modbus-tcp

sudo ufw allow 502-tcp

7.3 ssh

sudo ufw allow 22-tcp

7.4 django

sudo ufw allow 8000-tcp

7.5 redis

sudo ufw allow 6379-tcp

7.6 mqtt

sudo ufw allow 1883-tcp

sudo systemctl restart ufw

-Permisos a puerto especial 502

sudo apt update

sudo apt install libcap2-bin

sudo setcap 'cap\_net\_bind\_service=+ep' /usr/bin/python3.11

sudo getcap /usr/bin/python3.11

— Respuesta:

/usr/bin/python3.11 = cap\_net\_bind\_service+ep

-MQTT:

sudo apt install mosquitto mosquitto-clients

sudo systemctl enable mosquitto

sudo systemctl start mosquitto

-REDIS:

1. Instalación:

sudo apt update

sudo apt install redis-server

2. Activación:

sudo systemctl enable redis-server

sudo systemctl start redis-server

3. Comprobación:

redis-cli ping

--> PONG

4. Config avanzada:

sudo nano /etc/redis/redis.conf

-CONFIG estable USB:

/etc/udev/rules.d/

-DRIVER 485 CP210x (conversor RS-485-USB)

1. Comprobar presencia de driver en kernel:

sudo dmesg | grep cp210x

2. Otorgar permisos a grupo USER:

sudo usermod -aG dialout $USER

3. Otorgar permisos al puerto:

sudo chmod a+rw /dev/ttyUSB0 ---> /dev/ttyS3

-CONFIG ZIGBEE DIGI XBEE (pass-through uart)

1. Identificar puerto USB:

sudo ls -l /dev/ttyUSB\*

2. Otorgar permisos al puerto:

sudo chmod a+rw /dev/ttyUSB1 ---> \_port1 = "/dev/ttyS0"

sudo chmod a+rw /dev/ttyUSB2 ---> \_port2 = "/dev/ttyS1"

3. GPS /dev/ttyS2 🡪 /dev/ttyACM0

4 RS485 /dev/tty3 🡪 /dev/ttyS0

--HACER PERMANENTES LOS PUERTOS COM:

sudo nano /etc/udev/rules.d/99-usb-serial.rules

Lo más cómodo es abrir otra ventana SSH

Conectamos los dispositivos y configuramos de uno en uno para facilitar la tarea.  
  
Conectamos por ejemplo el Zigbee 1:

Escribimos en el terminal:

ls /dev

Comprobamos que puertos /dev/ttyUSB\* se crea. Supongamos /dev/ttyUSB0

Ver los dispositivos detectados:  
udevadm info -e | grep -A 20 'DEVNAME=/dev/tty'

El siguiente comando nos dará todos los valores del COM:

udevadm info --attribute-walk --path=/sys/bus/usb-serial/devices/ttyUSB0

Buscamos:  
  
ATTRS{idVendor}=="xxxx"

ATTRS{idProduct}=="xxxx"

ATTRS{serial}=="xxxx"

Añadimos al archive que generamos en la otra conexión SSH abierta y completamos con los valores. El archivo final “/etc/udev/rules.d/99-usb-serial.rules” tendrá la siguiente forma con los valores propios leídos de cada dispositivo:  
  
SUBSYSTEM=="tty", ATTRS{idVendor}=="0403", ATTRS{idProduct}=="6001", ATTRS{serial}=="215MOKNI", SYMLINK+="ZB1"

SUBSYSTEM=="tty", ATTRS{idVendor}=="0403", ATTRS{idProduct}=="6001", ATTRS{serial}=="215MOMCI", SYMLINK+="ZB2"

SUBSYSTEM=="tty", ATTRS{idVendor}=="10c4", ATTRS{idProduct}=="ea60", ATTRS{serial}=="0001", SYMLINK+="RS485"

En Ryzen PC-ind

Para renombrar serial (no USB)

RS485 🡪 /dev/ttyS0

GPS 🡪 /dev/ttyAMC0

sudo nano /etc/udev/rules.d/99-serial.rules

SUBSYSTEM=="tty", KERNEL=="ttyS0", SYMLINK+="RS485"

SUBSYSTEM=="tty", KERNEL=="ttyAMC0", SYMLINK+="GPS"

Comprobar:  
 ls -l /dev/RS485

ls -l /dev/GPS

Validamos cambios y reiniciamos

sudo udevadm control --reload-rules

sudo udevadm trigger

sudo reboot

-Python:

0. Establecer alias python=python3

nano ~/.bashrc

alias python='python3'

source ~/.bashrc

sudo apt install python3.11-venv

\*\*Crear entorno (necesario en debian 12)

python3 -m venv /mnt/sda1/tce/app/env/

\*\*Activar entorno

source /mnt/sda1/tce/app/env/bin/activate

\*\*Instalar librerias\*\*

Pip install -r requirements

-MIGRAR DB "tsm":

1. Copiar la db. Ej ruta /home/tc/data por sftp

2. Crear db:

2.1 Login en mariadb:

mysql -u root -p

2.2 Crear db "tsm":

MariaDB [(none)]> create database tsm;

2.3 Salir mariadb:

MariaDB [(none)]> exit;

2.3 Volcar db:

mysql -u root -p tsm < /home/tc/data/tsm.sql

-Hacer tc propietario de /mnt

sudo chown tc /mnt

-Desactivar hibernación:

sudo nano /etc/systemd/logind.conf

· Configuramos las opciones:

HandleLidSwitch=ignore

HandleLidSwitchDocked=ignore

HandleHibernateKey=ignore

HandleLidSwitchHibernate=ignore

sudo systemctl restart systemd-logind

MODIFICACIONES SOBRE FW:

1. urls.py (mod) -->> archivo en carpeta debian install

command = """

ip address show | grep -o -E '([[:xdigit:]]{1,2}:){5}[[:xdigit:]]{1,2}'

"""

2. /home/tc/app/evel/echoserver.py (mod) -->

2. bootlocal\_i3.sh (mod)

3. dos2unix bootlocal\_i3.sh

3. python manage.py set\_network (mod)

4. /home/tc/app/evel/echoserver.py (mod)

5. python3 manage.py runserver 192.168.0.235:8000

-CONFIGURAMOS EL AUTOARRANQUE DE DJANGO:

Crear rc.local:

sudo nano /etc/rc.local

Copiamos el siguiente contenido al archivo:

#!/etc/bash -e

# Activate the Python environment

source /mnt/sda1/tce/app/env/bin/activate

#Ir a ruta

cd /mnt/sda1/tce/app/app

# Launch the set network configuration from app

python manage.py set\_network > /dev/null 2>&1 &

# Uncomment the following line to enable DEBUG\_LOG

DEBUG\_LOG="true"

if [ -z "$DEBUG\_LOG" ]

then

python manage.py redis\_uart > /dev/null 2>&1 &

sleep 20

sudo daphne -b 0.0.0.0 -p 8000 tsm.asgi:application > /dev/null 2>&1 &

else

script -fqa -c "python3 manage.py redis\_uart" /mnt/sda1/logs/redis\_uart.log &

sleep 20

script -fqa -c "daphne -b 0.0.0.0 -p 8000 tsm.asgi:application" /mnt/sda1/logs/daphne.log &

fi

# Wait for the web server to be up and running

SERVER\_STATUS=`pgrep -f "/mnt/sda1/tce/app/env/bin/daphne"`

while [ -z "$SERVER\_STATUS" ]

do

sleep 1

SERVER\_STATUS=`pgrep -f "/mnt/sda1/tce/app/env/bin/daphne"`

done

# Launch other manage processes once the web server is running

if [ -z "$DEBUG\_LOG" ]

then

python manage.py sampler\_xbee > /dev/null 2>&1 &

python manage.py sampler\_weather > /dev/null 2>&1 &

python manage.py sampler\_tsm\_panel > /dev/null 2>&1 &

python manage.py sampler\_mqtt > /dev/null 2>&1 &

python manage.py start\_modbus > /dev/null 2>&1 &

python manage.py watchdog\_tsm > /dev/null 2>&1 &

else

script -fqa -c "python3 manage.py sampler\_xbee" /mnt/sda1/logs/sampler\_xbee.log &

script -fqa -c "python3 manage.py sampler\_weather" /mnt/sda1/logs/sampler\_weather.log &

script -fqa -c "python3 manage.py sampler\_tsm\_panel" /mnt/sda1/logs/sampler\_tsm\_panel.log &

script -fqa -c "python3 manage.py sampler\_mqtt" /mnt/sda1/logs/sampler\_mqtt.log &

script -fqa -c "python3 manage.py start\_modbus" /mnt/sda1/logs/start\_modbus.log &

script -fqa -c "python3 manage.py watchdog\_tsm" /mnt/sda1/logs/start\_modbus.log &

fi

exit 0

LANZAMOS DE CRONTAB EL ARCHIVO EN EL INICIO DEL SISTEMA

2. crontab -e

@reboot bash /etc/rc.local

CONFIGURAR ZB CON SW DIGI XCTU:

Baud Rate: BD 🡪115200 (7)

API Mode: AP 🡪 API Mode Without Scapes (1)

Device Role: CE 🡪 Form Network (1)

Encryption Enable: EE 🡪 Enabled (1)

Link Key KY 🡪 52ABD862B129000EBF970135B0758BD5

MODIFCACIONES DE Puerto RS485 en FW:

A) \app\libs\_modifications\modbus\_tk-1.1.1\modbus\_rtu.py

140,43: if self.\_serial.name == '/dev/RS485':

154,43: if self.\_serial.name == '/dev/RS485':

B) \app\tsm\singletons\singleton\_uart.py

223,64: self.master = modbus\_rtu.RtuMaster(serial.Serial("/dev/RS485", baudrate= 9600, bytesize=8, parity="N", stopbits=1))